2INC0 - Operating Systems

Course details

Dr. Geoffrey Nelissen

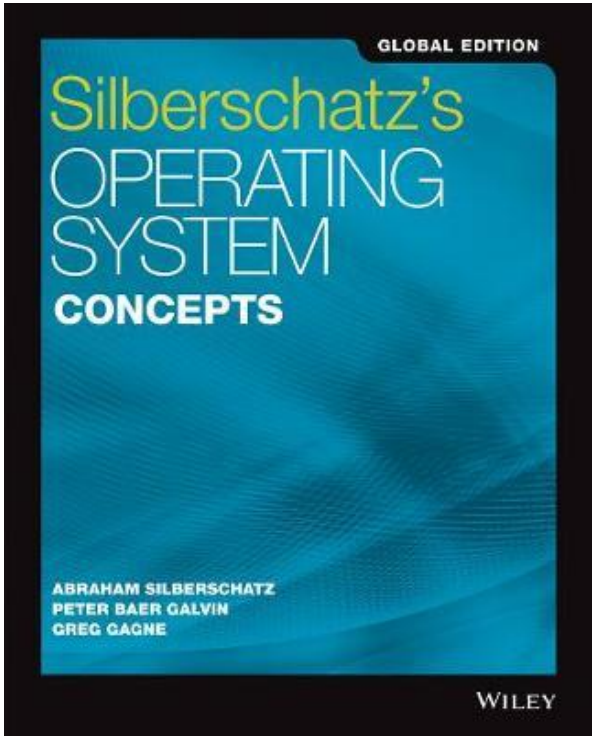
IRiS
Interconnected
Resource-aware
Intelligent Systems

TU/e
Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

Course organization

- **Lectures**
  - **Wednesday** **13:30 – 15:15**
  - **Friday** **8:45 – 10:30**
- **Instruction sessions**
  - **Wednesday** **15:30 – 17:15**
  - **Friday** **10:45 – 12:30**



- **Textbook**

  "Operating System Concepts" by Abraham Silberschatz, Peter B. Galvin, Greg Gagne, (2020 - 10th edition - Global edition) *ISBN: 9781119454083*

Course organization

- **Lectures**

- **Homework assignments** (5 or 6 **quizzes on Canvas**, weekly)
  - strict deadline (see Canvas).
  - You will always have at least one week to do the homework.

- **Practical assignments** (3 in total)
  - handed in with a report containing discussion and results
  - submission through Canvas before deadline
    - score := max (0, score - #days_overdue)
    - Resubmissions after deadline are discarded since grading has already started.
  - Linux based assignments
  - Done in groups of **3 students**

- **Instruction hours (every week)**
  - help with practical assignments
  - help with Linux and C programming
  - working on exercises, homework, etc.

New this year:
Workshops to learn C and POSIX

How will you be evaluated?

- final exam (written) 50% (min 5.0 to pass)
- practical 40%
- homework average 10%

Who is involved?

- **Lectures**
  - **Geoffrey Nelissen**
  - Mitra Nasri


- **Workshops**
  - **Olav Bunte**

- **Practical assignments**
  - **Olav Bunte** (**Main instructor**)
  - Saeed Khalilian Gourtani
  - Ali Assi
  - Iliyan Teofilov
  - Bhuvan Vankadari
  - Mir Yousif

Pre-knowledge (1)

- **Basic knowledge of computer building blocks**
  - notions of **ALUs**, **registers**, **data paths**, control units, **memory**, input & output, **interrupts** (as treated in e.g. 2IC30).

If you do not have the pre-knowledge, **study the slides sets provided on Canvas**

Pre-knowledge (2)

- **Programming skills**
  - We assume that you have reasonable programming experience, e.g. in Java, C, C++, Rust.
  - **The assignments will be written in C in Linux.**

Note: New this year **based on student's feedback.**
We provide **4 workshops** to teach the **basics of C** **and the part of the POSIX API you'll need for the assignment**

Workshops schedule:
- Wednesday – Nov. 13
- Friday – Nov. 15
- Friday – Dec. 6
- To be fixed: Friday – Dec. 13 **or** Wednesday – Dec. 18

The practical assignments

- Announcement and submission are through Canvas.
- **Join a group** on Canvas by Friday

> **Note: Students who already attended** the course in a previous year **cannot join a group** with **students attending for the first time** in 2024

- We will provide a C-template per assignment to make sure that you do not have to start from scratch.
- Follow the **workshops**. They **will give you a head start.**
- Start the practical assignments on time or you will have difficulties to finish them before the deadline.

Exemptions:
If you passed the assignment last year and want to carry over your grade, register **in Canvas**

Weekly homework

Note: the **type of questions** in the homework assignments **changed w.r.t. two years ago following students' feedback**. It became **more formative**.

- Weekly Canvas quizzes that you do at home
- **Goals**:
  - avoid that you fall behind
  - Prepare you for part of the exam
- Each quiz covers a different part of the course
  Covered material is mentioned on the welcome page of each quiz
- Submissions and grading are electronic via Canvas.
  - *We use a test bank for each homework (many questions/exercises).*
    - *A student is assigned a random set of questions for each homework.*
    - *Average difficulty of questions in each homework for each student is the same.*

Learning objectives (1)

At the end of the course, the student is able to:

O1. **explain the relevant concepts related to operating systems** such as transparency, virtualization, threads, processes, scheduling, I/O management, etc.;

O2. **identify problems** that may impact the **correctness of** example **programs executed in a concurrent execution environment**;

O3. **synchronize the execution of several threads and processes** using semaphores, condition variables and monitors;

Learning objectives

At the end of the course, the student is able to:

O4. **analyze synchronization and communication protocols** between threads and processes to check whether they respect invariants and avoid deadlocks;

O5. **apply memory and file system management techniques** on example systems.

O6. **develop concurrent programs on Linux** using **C** and the **POSIX** API.

Course Overview

- **Introduction to operating systems** (lecture 1)
- **Processes, threads and scheduling** (lectures 2 and 3)
- **Concurrency and synchronization**
  - atomicity and interference (lecture 4)
  - action synchronization (lecture 5)
  - condition synchronization (lecture 6)
  - deadlock (lecture 7)
- **File systems** (lectures 8)
- **Memory management** (lectures 9 and 10)
- **Input/output** (lecture11)

Tentative schedule available on Canvas

How to study?

- **Prepare before each lecture** by studying the preparatory material referenced on Canvas
  (short videos and/or a few sections of the reference book)

- **Do the weekly homework exercises.**

- **Come to the lectures:**
  - Lectures will cover details that are not in the preparatory material.
  - There will be **exercises and quizzes during the lectures** to prepare you for the exam and the homework

- **Go to the workshops**

Announcements

You **cannot register** for 2INC0 **in Q3 if you follow it in Q2**

The **resit** of 2INC0 will be **in the afternoon,** not the usual evening

Fraud is not permitted!

- You have signed the scientific code of conduct document and you are LEGALLY bound to it. Any form of **fraud and/or suspicion of fraud will be reported to the exam committee** and will have (severe) consequences. In particular, but not exclusively, you should be aware of:
  - **Anything you hand in MUST be your own work** and MUST contain a **proper citation** of external resources (if and when they are used).
  - You are NOT allowed to **copy/use text/figures/pictures** from publications or the Internet and **work/code** found on repositories **(e.g. github, gitlab, stack overflow, studeersnel) without a proper citation.**
  - You are NOT allowed to **copy work of others**, e.g., your fellow students, or hand in work of others as your own.
  - You are NOT allowed to **make your work available** (in any form) to fellow students.
  - You MUST keep your course related Internet **repositories PRIVATE.** This holds **during and after the course** (even after your graduation!).

- Must read references:
  - (i) https://www.tue.nl/en/our-university/about-the-university/integrity/
  - (ii) https://educationguide.tue.nl/practical-info/regulations-codes-of-conduct-and-guidelines/

Beware of breach of copyright!

- Course learning materials are Eindhoven University of Technology's copyright and you are **NOT allowed to publish any copyrighted material** of third parties in any form on Internet. In particular, but not exclusively, you should be aware of:
  - You are NOT allowed to make any **part of the course (including exams, answers to exam questions, practical/lab assignments or their solutions) available on Internet and/or repositories (e.g. github, gitlab, stack overflow, studeersnel).** This holds **during and after the course** (even after your graduation!).

- Must read reference: https://www.tue.nl/en/our-university/about-the-university/integrity/

Let's see if you understood how it works

Just to break the ice, first question:

From **which movie** is this one of the main characters?

Let’s see if you understood how it works

Just to break the ice, first question:

From which movie is this one of the main characters?